

# **SVM – Support Vector Machines**

## **Metoda wektorów nośnych**

# Plan wykładu

---

1. Liniowa separowalność w statystycznej klasyfikacji
2. Podstawy metody SVM
3. Uogólnienie SVM (nie w pełni separowalne liniowo)
4. Funkcje jądrowe (kernel functions)
5. SVM dla danych separowalnych nieliniowo
6. Podsumowanie
7. Gdzie szukać więcej

# Formalizacja problemu klasyfikacji

- W przestrzeni danych (ang. measurement space)  $\Omega$  znajdują się wektory danych  $\mathbf{x}$  stanowiące próbkę uczącą  $D$ , należące do dwóch klas

$$D = \left\{ (\mathbf{x}_i, c_i) \mid \mathbf{x}_i \in R^p, c_i \in \{1, -1\} \right\}_{i=1}^N$$

- Szukamy klasyfikatora pozwalającego na podział całej przestrzeni  $\Omega$  na dwa rozłączne obszary odpowiadające klasom  $\{1, -1\}$  oraz pozwalającego jak najlepiej klasyfikować nowe obiekty  $\mathbf{x}$  do klas
- Podejście opiera się na znalezieniu tzw. granicy decyzyjnej między klasami  $\rightarrow g(\mathbf{x})$

# Separowalność liniowa

---

- Dwie klasy są liniowo separowalne, jeśli istnieje hiperpłaszczyzna  $H$  postaci  $g(\mathbf{x})$

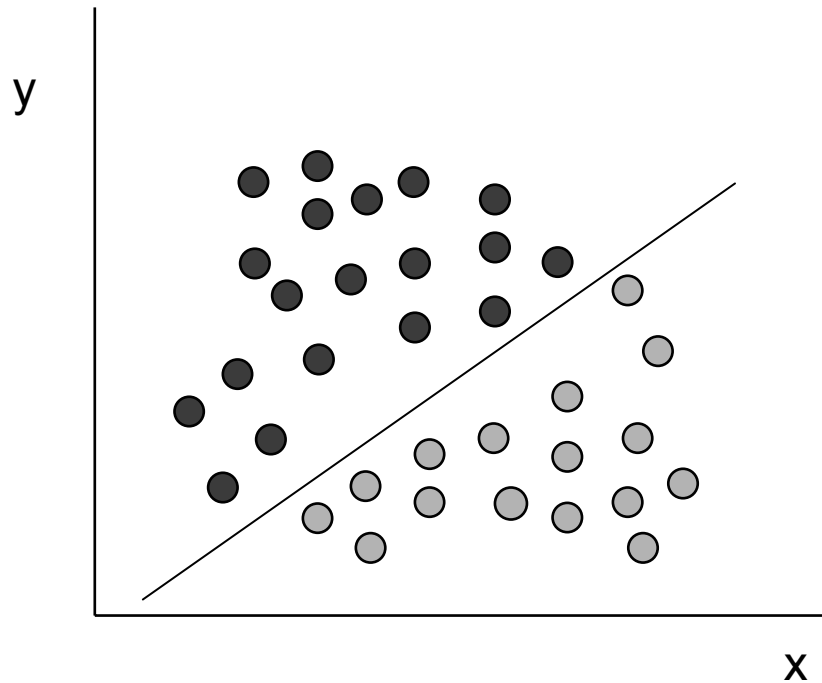
$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

- przyjmująca wartości

$$\begin{cases} g(\mathbf{x}_i) > 0 & \mathbf{x}_i \in 1 \\ g(\mathbf{x}_i) < 0 & \mathbf{x}_i \in -1 \end{cases}$$

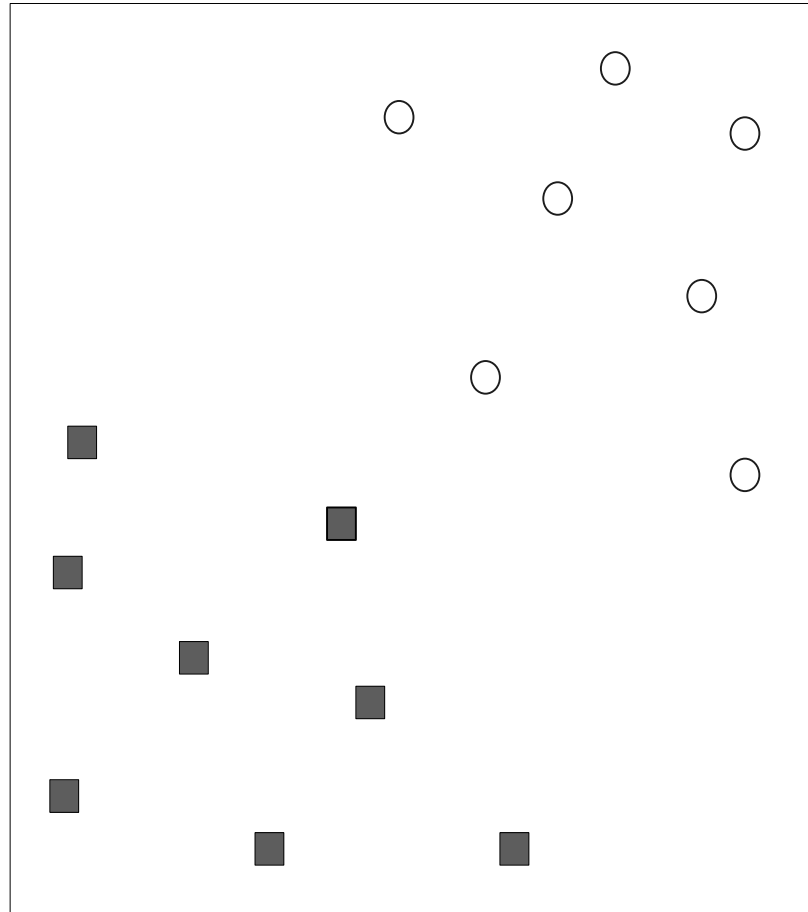
- Jak poszukiwać takiej hiperpłaszczyzny granicznej?

# Liniowa funkcja separująca



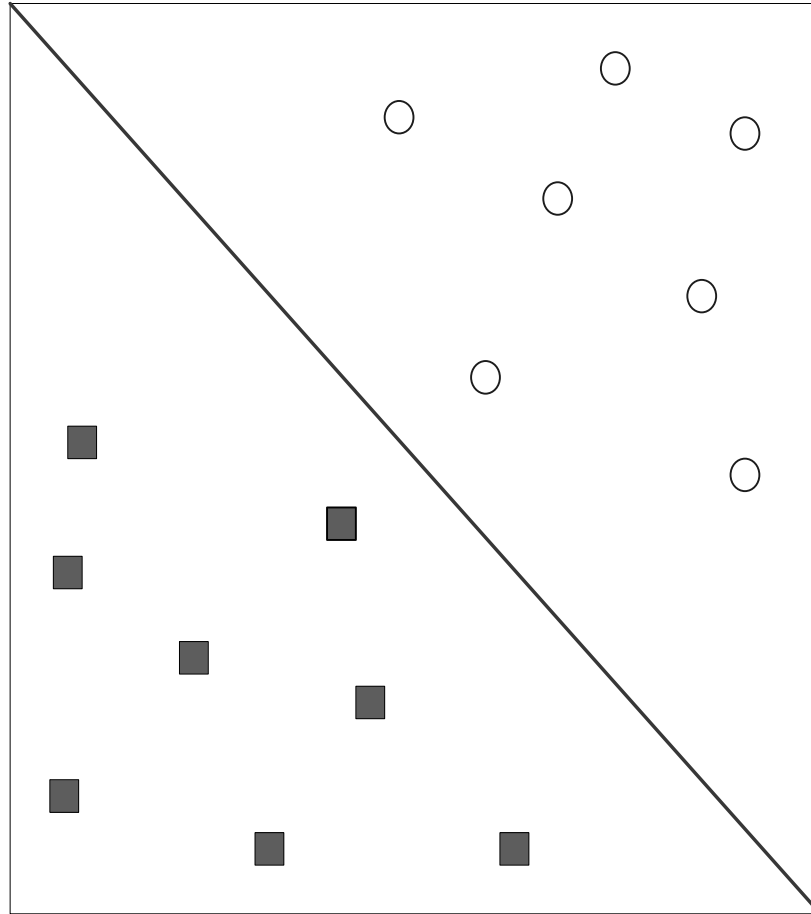
- Funkcja liniowa separująca
- Wyznacza podział przestrzeni na obszary odpowiadające dwóm klasom decyzyjnym.
- Oryginalna propozycja Fisher, ale także inne metody (perceptron, itp..)
- Uogólnienia dla wielu klas.

# Support Vector Machines



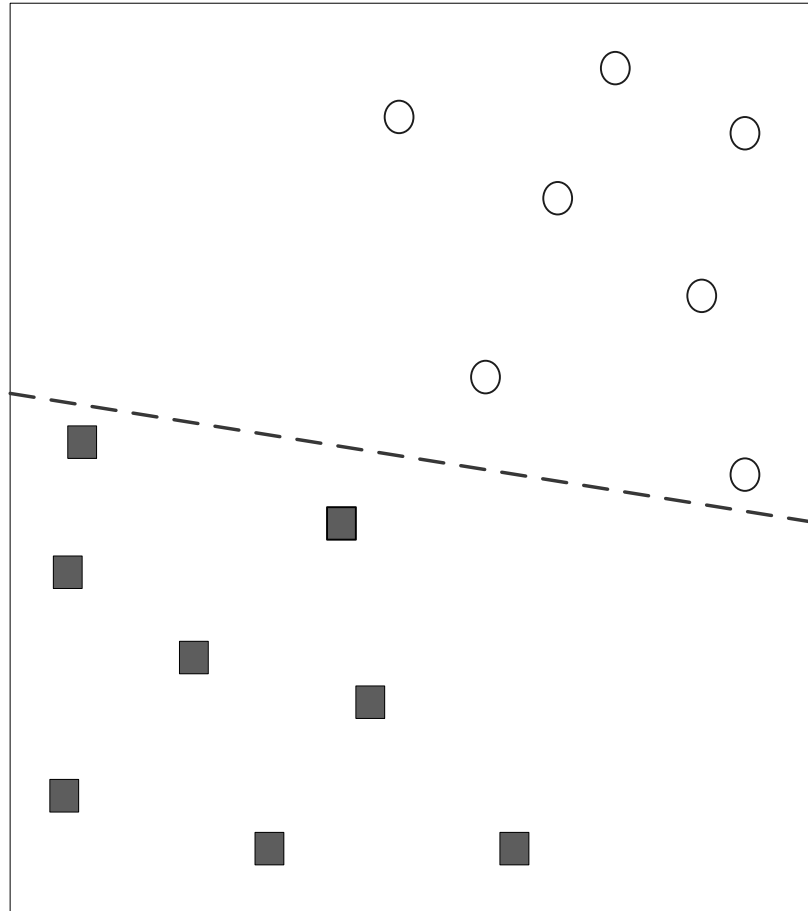
- Znajdź liniową hiperpłaszczyznę (decision boundary) oddzielającą obszary przykładów z dwóch różnych klas

# Support Vector Machines



- Jedno z możliwych rozwiązań

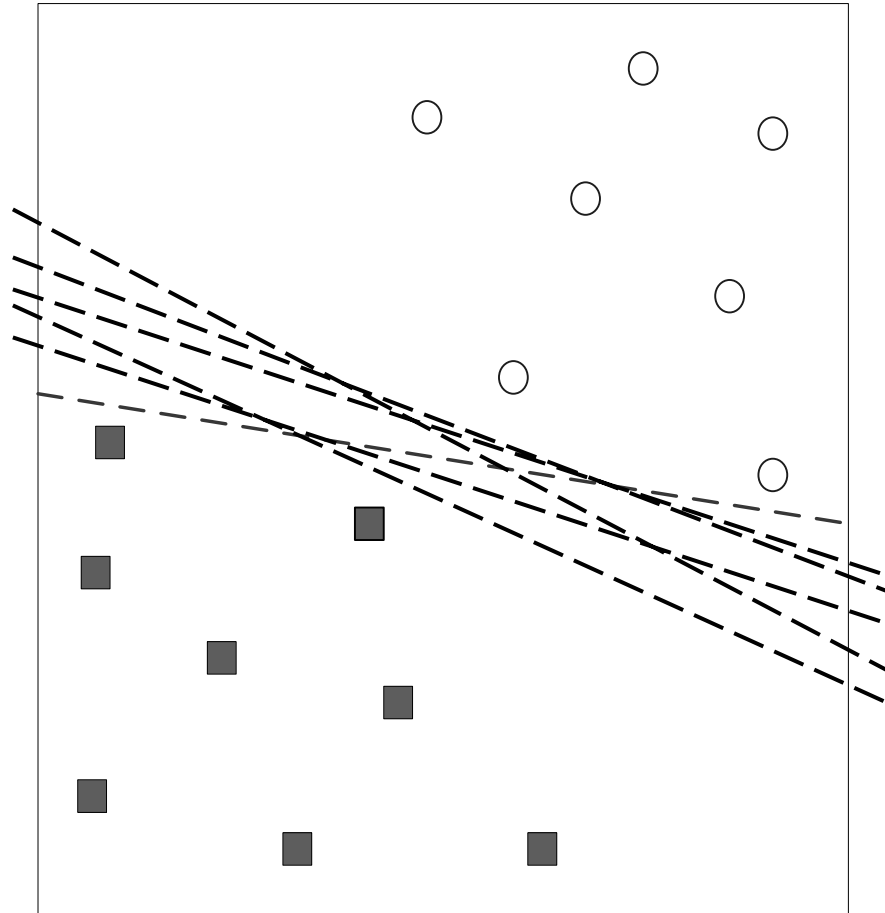
# Support Vector Machines



- Inne możliwe rozwiązanie

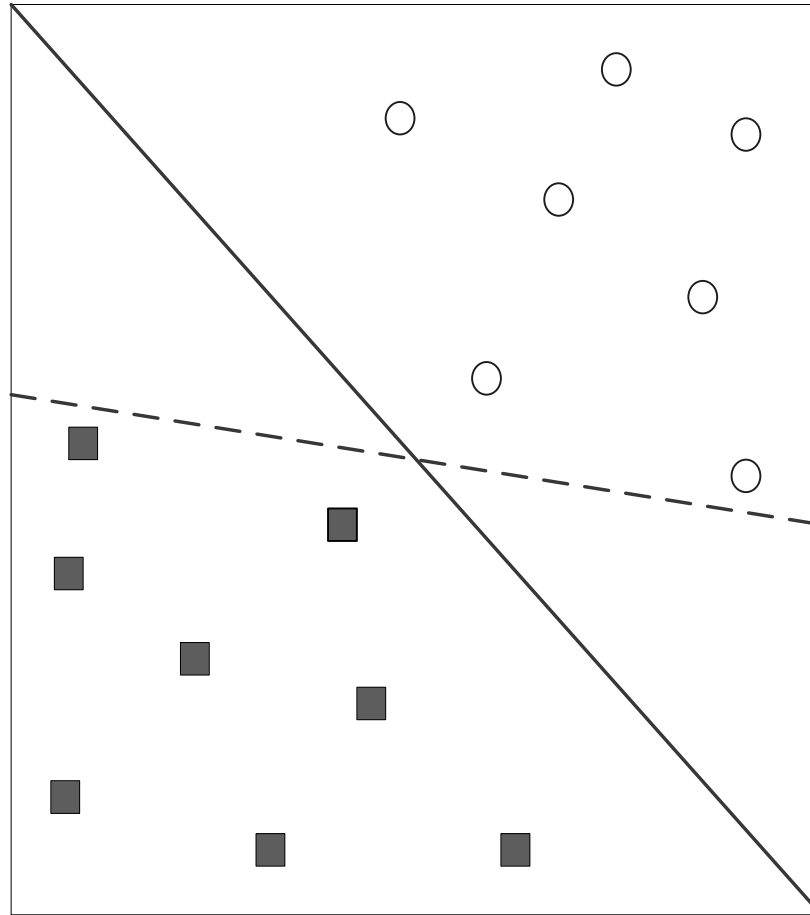


# Support Vector Machines



- Zbiór wielu możliwych rozwiązań

# Support Vector Machines



- Którą z hiperpłaszczyzn należy wybrać? B1 or B2?
- Czy można to formalnie zdefiniować?

# Uwagi o marginesie

- Hiperpłaszczyzny  $b_{i1}$  i  $b_{i2}$  są otrzymane przez równoległe przesuwanie hiperpłaszczyzny granicznej aż do pierwszych punktów z obu klas.
- Odległość między nimi – **margines** klasyfikatora liniowego
- Jaki margines wybierać?

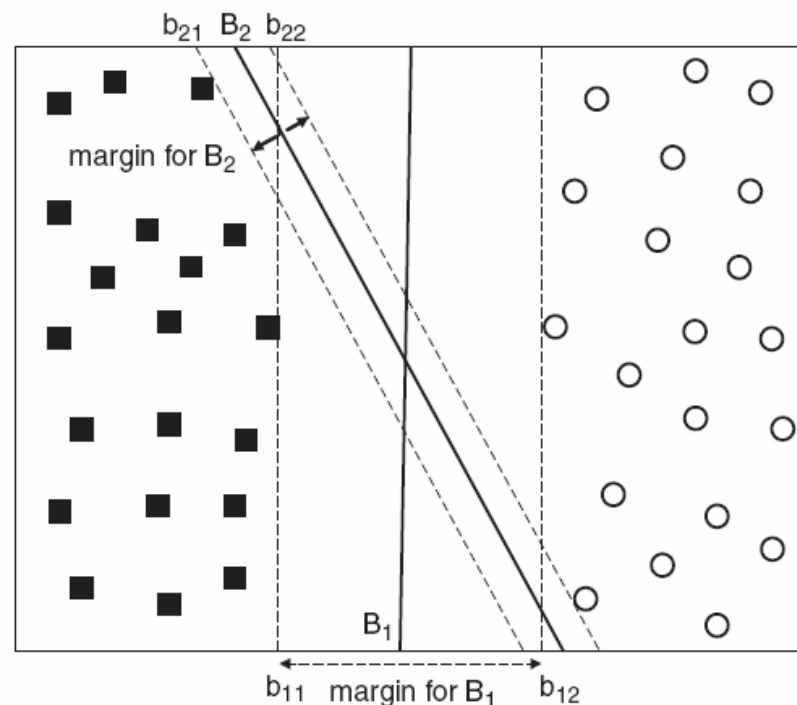
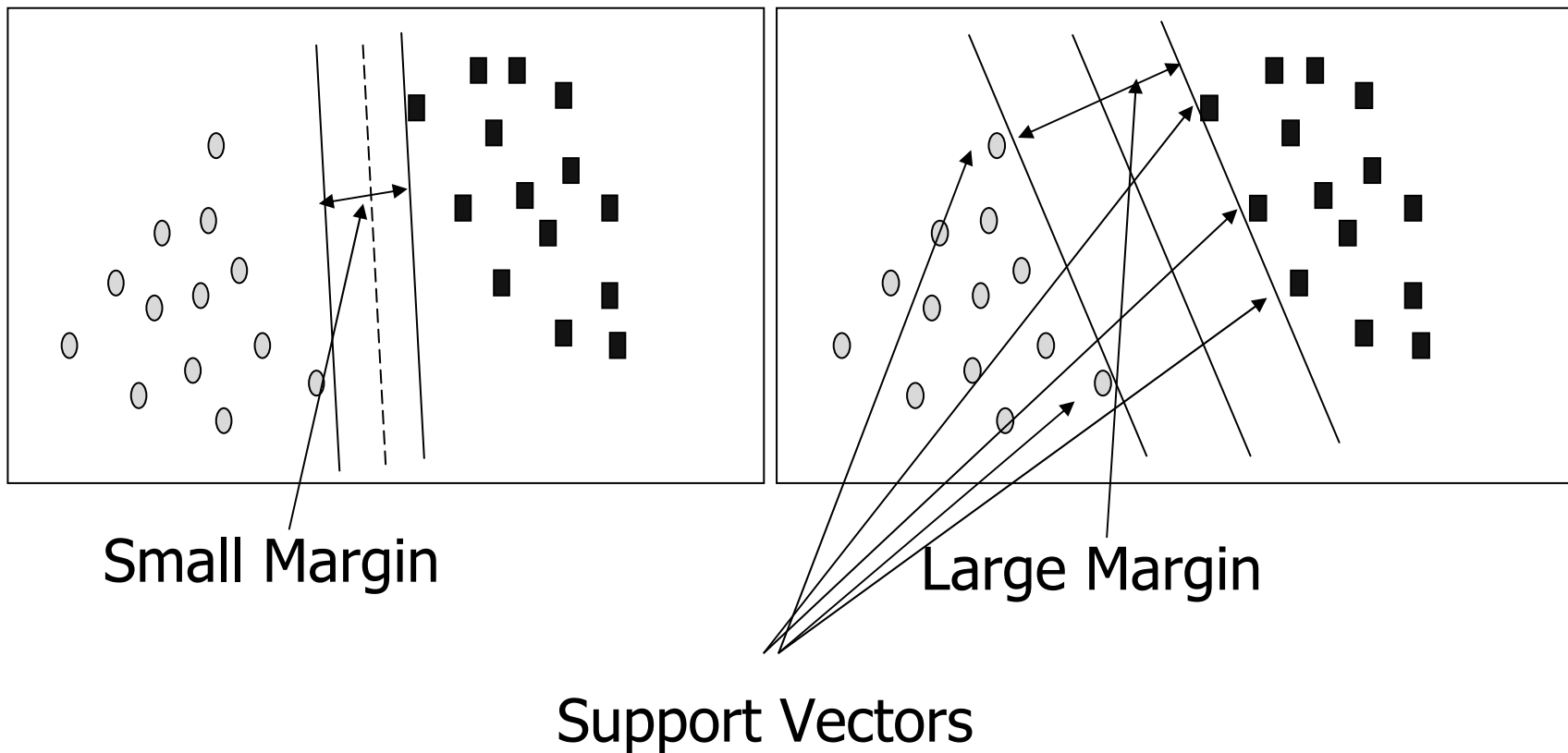


Figure 5.22. Margin of a decision boundary.

# Węższe czy szersze marginesy?

- Szerszy margines → lepsze własności generalizacji, mniejsza podatność na ew. przeuczenie (overfitting)
- Wąski margines – mała zmiana granicy, radykalne zmiany klasyfikacji



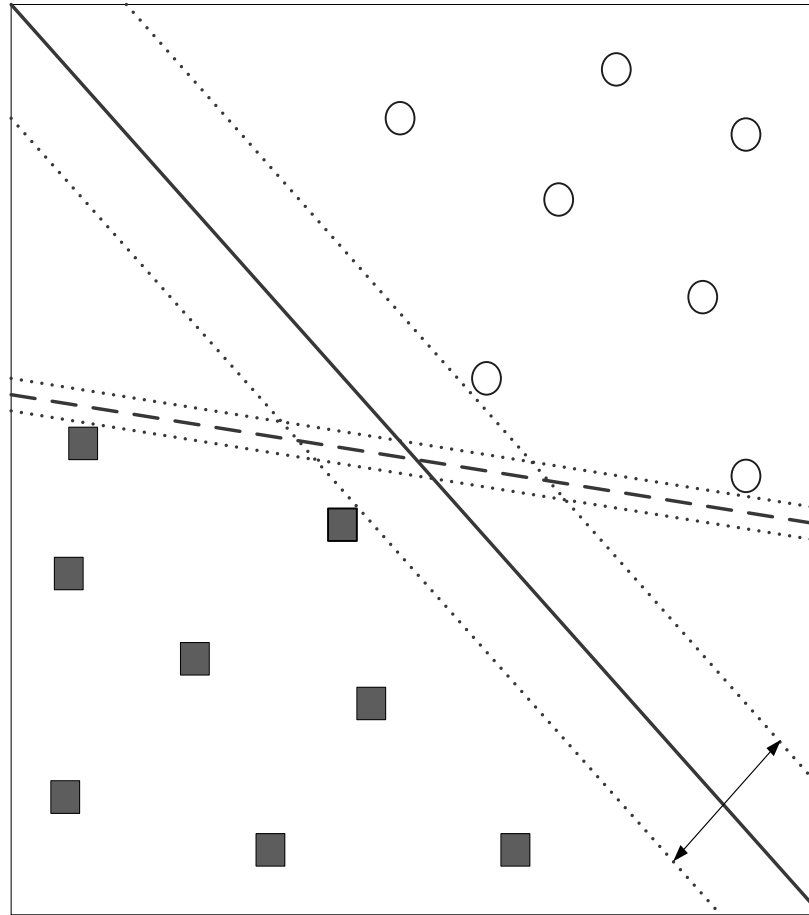
# Teoria „Structural risk minimization”

- Oszacowanie górnej granicy błędu ze względu na błąd uczący  $R_e$ , liczbę przykładów  $N$  i tzw. model complexity  $h$  z prawdopodobieństwem  $1-\eta$  „generalization error” nie przekroczy:

$$R \leq R_e + \varphi\left(\frac{h}{N}, \frac{\log(\eta)}{N}\right)$$

- Prace teoretyczne –  $h$  complexity dla modelu liniowego:
- „Models with small margins have higher capacity -complexity because they are more flexivle and can fit many training sets”
- Także „The hypothesis space with minimal VC-dimension according to SRM”
- Reasumując modele o większej complexity mają gorsze oszacowanie błędu
- Dlatego wybieraj większy margines!

# Support Vector Machines



- Znajdź hiperpłaszczyznę, która maksymalizuje tzw. **margins** => B1 jest lepsze niż B2

# Liniowe SVM hiperpłaszczyzna graniczna

- Vapnik – poszukuj „maximal margin classifier”

$$\mathbf{w} \cdot \mathbf{x} + \mathbf{b} = 0$$

gdzie  $\mathbf{w}$  i  $\mathbf{b}$  są parametrami modelu

$$y = \begin{cases} 1 & \mathbf{w} \cdot \mathbf{x} + \mathbf{b} > 0 \\ -1 & \mathbf{w} \cdot \mathbf{x} + \mathbf{b} < 0 \end{cases}$$

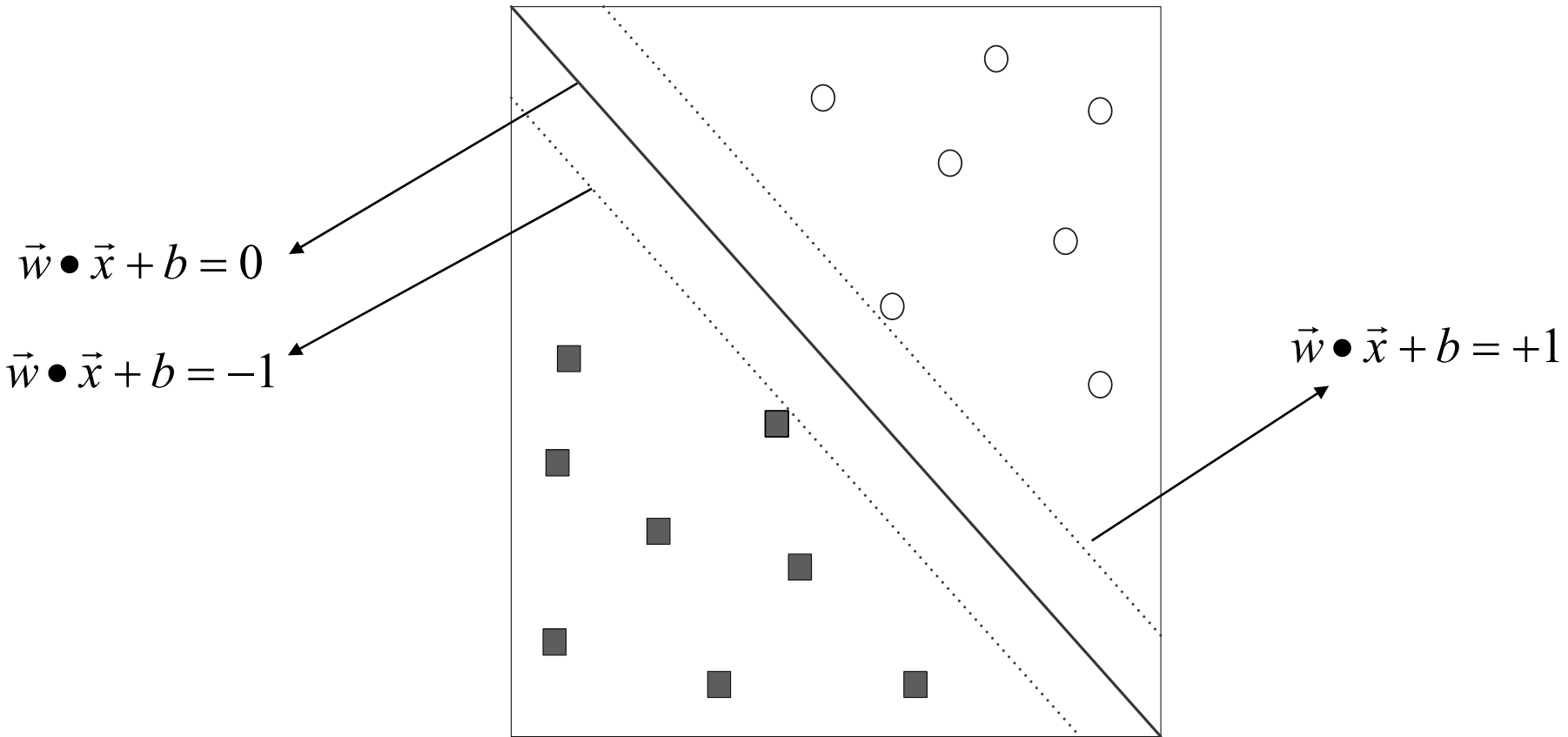
- Parametry granicy wyznaczaj tak, aby maksymalne marginesy  $b_{i1}$  i  $b_{i2}$  były miejscem geometrycznym punktów  $\mathbf{x}$  spełniających warunki

$$b_{i1} \quad \mathbf{w} \cdot \mathbf{x} + \mathbf{b} = 1$$

$$b_{i2} \quad \mathbf{w} \cdot \mathbf{x} + \mathbf{b} = -1$$

- Margines – odległość między płaszczyznami  $b_{i1}$  i  $b_{i2}$

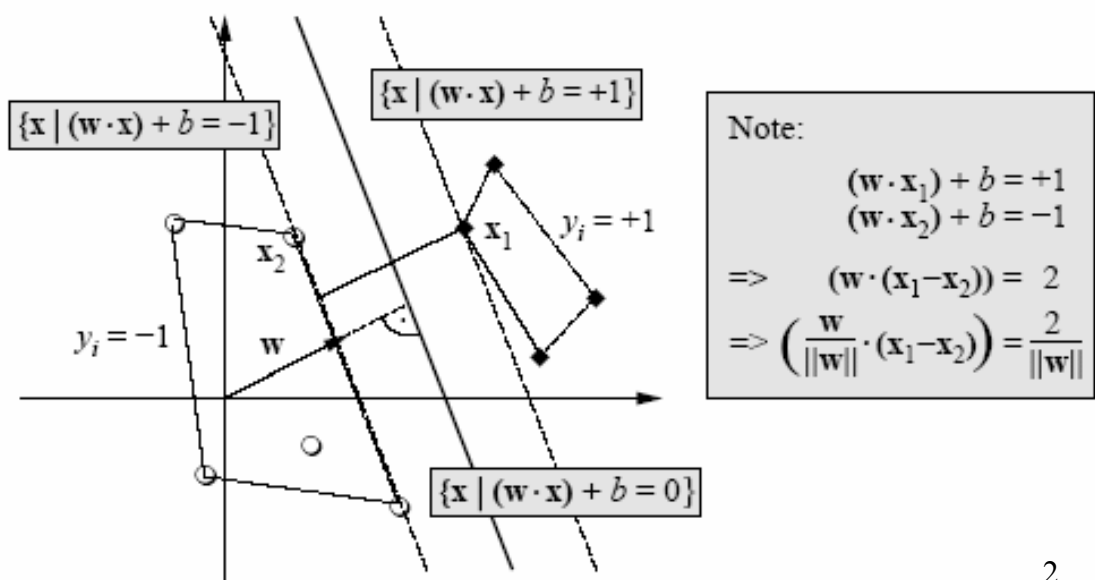
# Poszukiwanie parametrów hiperpłaszczyzny



$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$



# Ilustracje i sposób przekształceń



**Fig. 2.** A binary classification toy problem: separate balls from diamonds. The *optimal hyperplane* is orthogonal to the shortest line connecting the convex hulls of the two classes (dotted), and intersects it half-way between the two classes. The problem is separable, so there exists a weight vector  $w$  and a threshold  $b$  such that  $y_i \cdot ((w \cdot x_i) + b) > 0$  ( $i = 1, \dots, m$ ). Rescaling  $w$  and  $b$  such that the point(s) closest to the hyperplane satisfy  $|(w \cdot x_i) + b| = 1$ , we obtain a *canonical form*  $(w, b)$  of the hyperplane, satisfying  $y_i \cdot ((w \cdot x_i) + b) \geq 1$ . Note that in this case, the *margin*, measured perpendicularly to the hyperplane, equals  $2/\|w\|$ . This can be seen by considering two points  $x_1, x_2$  on opposite sides of the margin, i.e.,  $(w \cdot x_1) + b = 1$ ,  $(w \cdot x_2) + b = -1$ , and projecting them onto the hyperplane normal vector  $w/\|w\|$  (from [29]).

$$\text{margin} = \frac{2}{\|w\|}$$

$$\|w\| \equiv \sqrt{w_1^2 + \dots + w_p^2}$$

Cel: Maksymalizuj margines!

$$\frac{2}{\|w\|}$$

**maximize**

→

$$\frac{\|w\|}{2}$$

**minimize**

→

$$\frac{\|w\|^2}{2}$$

**minimize**

# Linear Support Vector Machines

---

- Sformułowanie mat. problemu:

$$\min_{\mathbf{w}} = \frac{\|\mathbf{w}\|^2}{2}$$

- Przy warunkach ograniczających

$$y_i(w \cdot x_i + b) \geq 1 \quad i = 1, 2, \dots, N$$

- Jest to problem optymalizacji kwadratowej z liniowymi ogr. → uogólnione zadanie optymalizacji rozwiązywany metodą mnożników Lagrange'a (tak aby np. nie dojść do  $w \rightarrow 0$ )

# LSVM

- Minimalizuj funkcję Lagrange'a

$$L(w, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^N \alpha_i (y_i (\mathbf{w} \mathbf{x}_i + b) - 1)$$

- parametry  $\alpha \geq 0$  mnożniki Lagrange'a
- Powinno się różniczkować  $L$  po  $w$  i  $b$  – nadal trudności w rozwiązaniu
- Przy przekształceniach wykorzystuje się ograniczenia Karush-Kuhn-Tucker na mnożniki:

$$\alpha_i \geq 0$$

$$\alpha_i [y_i (w \cdot x_i + b) - 1] = 0$$

- W konsekwencji  $\alpha_i$  są niezerowe wyłącznie dla wektorów nośnych  $\mathbf{x}$ , pozostałe są zerowe
- Rozwiązanie parametrów  $\mathbf{w}$  i  $b$  zależy wyłącznie od wektorów nośnych.

# LSVM – sformułowanie dualne

- Nadal zbyt wiele parametrów  $w, b, \alpha$  do oszacowania
- Przechodzi się na postać dualną zadania optymalizacji

- Maksymalizuj  $L(\alpha)$ 
$$\sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

- Przy ograniczeniach

$$\alpha_i \geq 0, \quad \forall i \quad \sum_{i=1}^N \alpha_i y_i = 0$$

- Rozwiązanie ( $\alpha > 0$  dla  $i \in \text{SV}$ ) ;  $b$  – odpowiednio uśredniane

$$\mathbf{w} = \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i$$

- Hiperpłaszczyzna decyzyjna

$$\sum_{i=1}^N \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b = 0$$

# Rozwiązanie LSVM

- Klasyfikacja – funkcja decyzyjna

$$f(x) = \text{sign}\left(\sum_{i=1}^N \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b\right)$$

- O ostatecznej postaci hiperpłaszczyzny decydują wyłącznie wektory nośne ( $\alpha_i > 0$ )
- Im większa wartość  $\alpha_i$  tym większy wpływ wektora na granicę decyzyjną
- Klasyfikacja zależy od iloczynu skalarnego nowego  $\mathbf{x}$  z wektorami nośnymi  $\mathbf{x}_i$  ze zbioru uczącego
- Pewne założenie metody – starać się zbudować klasyfikator liniowy używając możliwie minimalną liczbę wektorów z danych treningowych (wektory nośne)
- Funkcjonalnie klasyfikator podobny do niektórych sieci neuronowej, metod jądrowych Parzena

# Przykład

Obliczmy wagi

$$w_1 = \sum_i \alpha_i y_i x_{i1} = 65.5621 \cdot 1 \cdot 0.3858 + 65.5621 \cdot (-1) \cdot 0.4871 = -6.64$$

$$w_2 = \sum_i \alpha_i y_i x_{i2} = 65.5621 \cdot 1 \cdot 0.4687 + 65.5621 \cdot (-1) \cdot 0.611 = -9.32$$

$$b' = 1 - (-6.64) \cdot 0.3858 - (-9.32)(0.4687) = 7.930$$

$$b'' = -1 - (-6.64) \cdot 0.4871 - (-9.32)(0.611) = 7.928$$

$$b = 0.5 \cdot (b' + b'') = 7.93$$

$x_1$	$x_2$	$y$	Lagrange Multiplier
0.3858	0.4687	1	65.5261
0.4871	0.611	-1	65.5261
0.9218	0.4103	-1	0
0.7382	0.8936	-1	0
0.1763	0.0579	1	0
0.4057	0.3529	1	0
0.9355	0.8132	-1	0
0.2146	0.0099	1	0

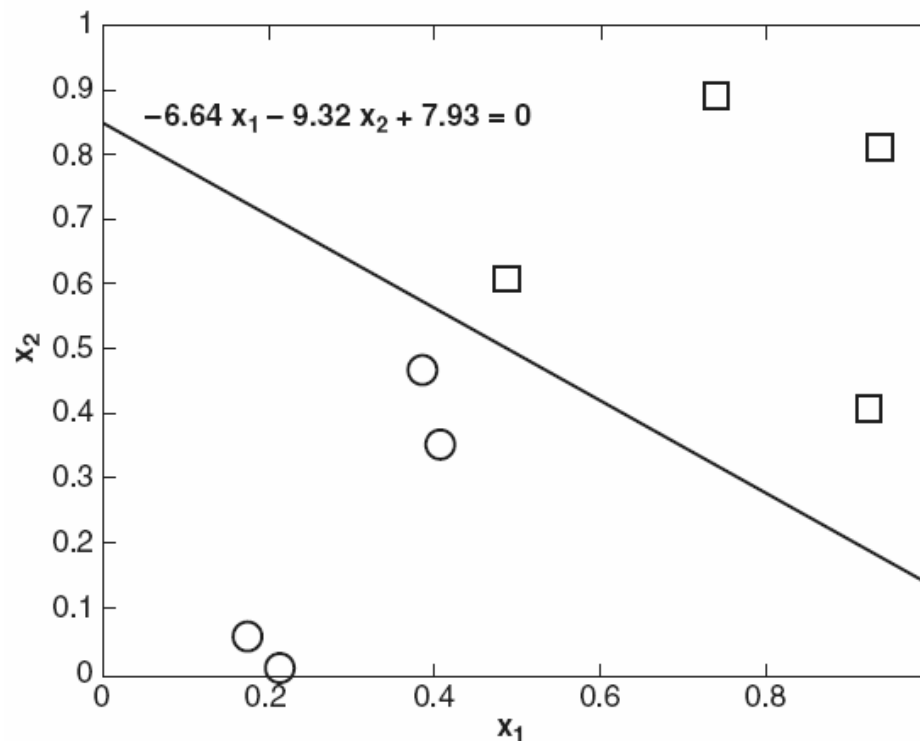
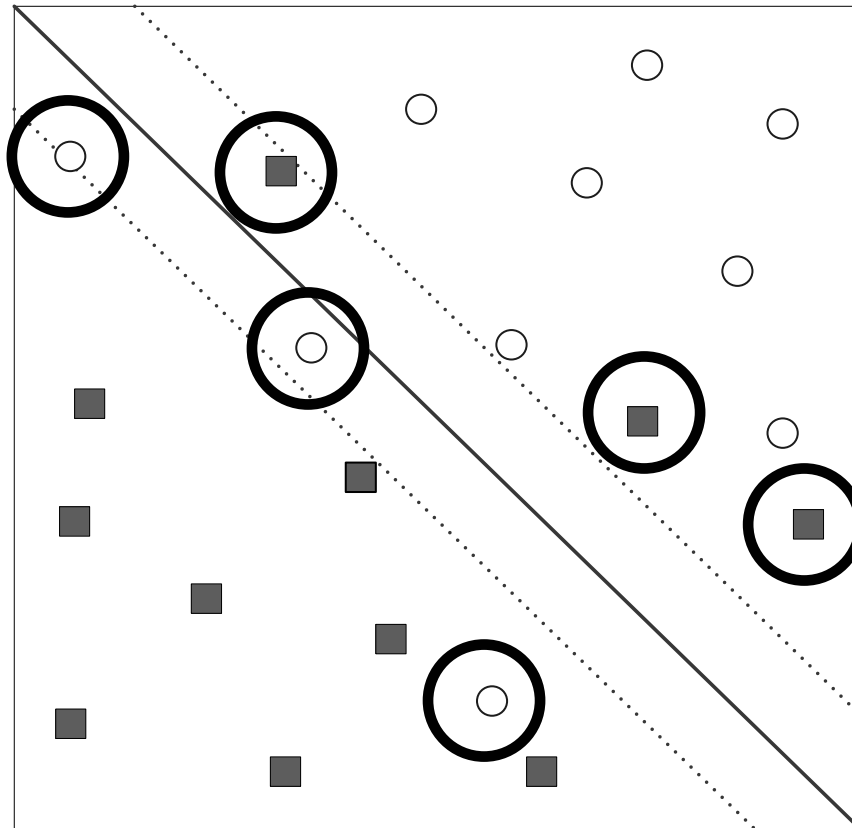


Figure 5.24. Example of a linearly separable data set.

# Support Vector Machines

- Co robić z LSVM gdy dane nie są w pełni liniowo separowalne?



# Jak użyć SVM dla dane liniowo nieseparowalnych

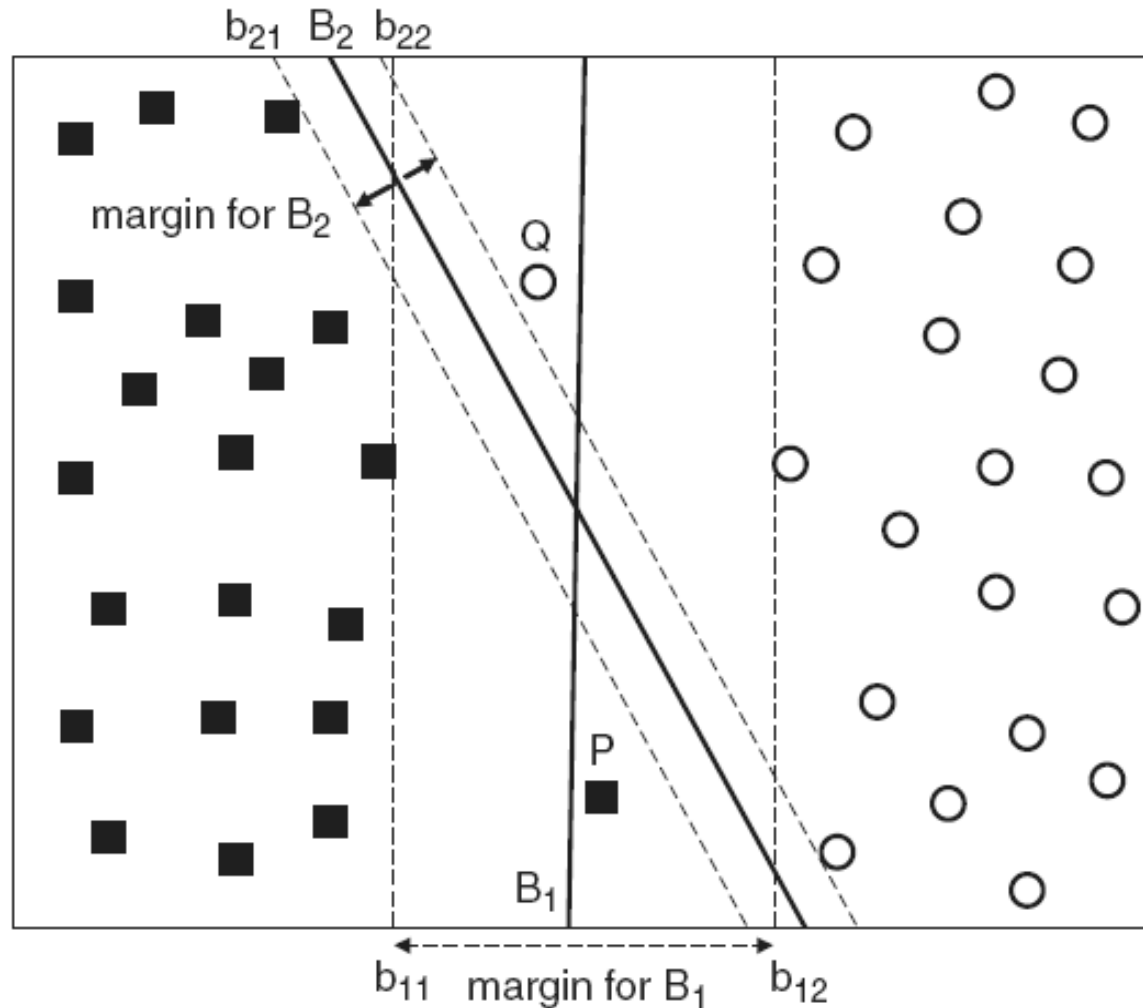


Figure 5.25. Decision boundary of SVM for the nonseparable case.



# Zmienne dopełniające

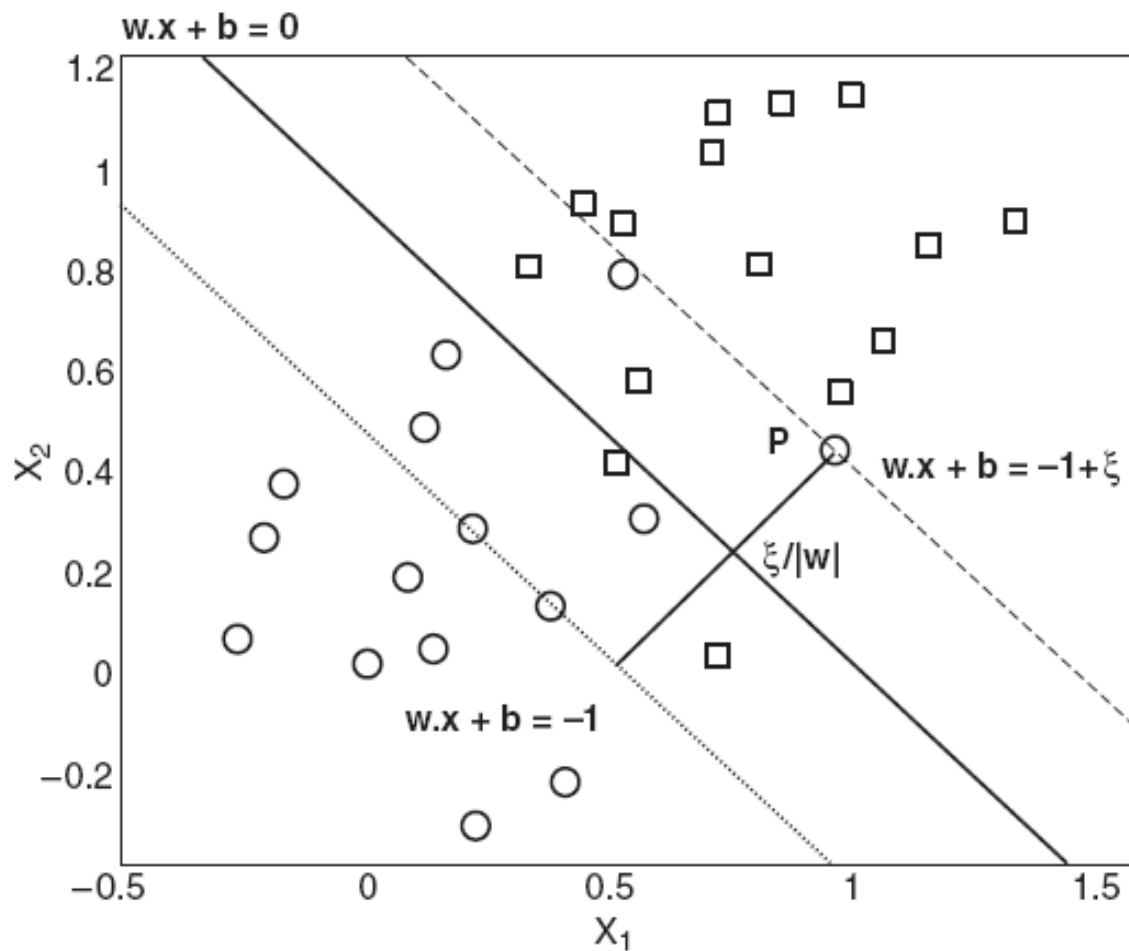


Figure 5.26. Slack variables for nonseparable data.

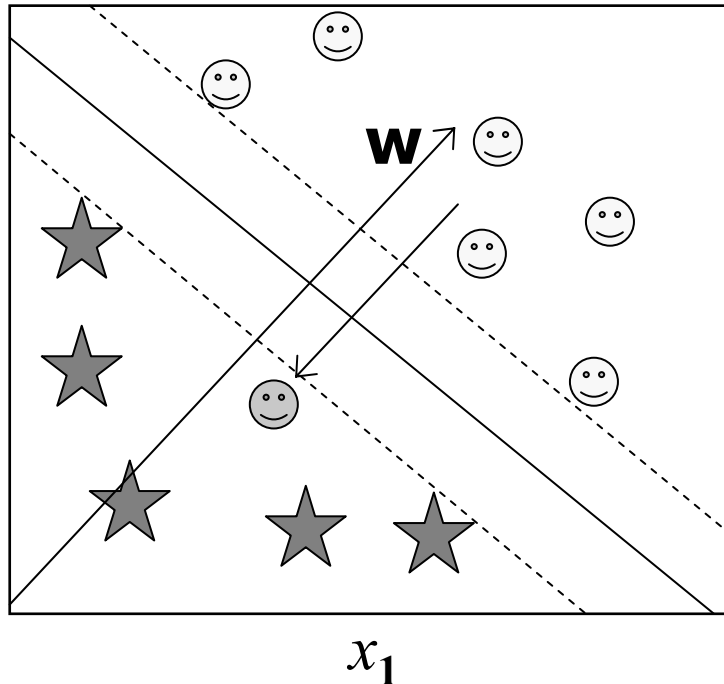
# Zmienne osłabiające - interpretacja

- Zmienne  $\xi_i \geq 0$  dobiera się dla każdego przykładu uczącego. Jej wartość zmniejsza margines separacji. (rodzaj „zwisu” punktu poza hiperpłaszczyzną nośną)
- Jeżeli  $0 \leq \xi_i \leq 1$ , to punkt danych  $(\mathbf{x}_i, d_i)$  leży wewnątrz strefy separacji, ale po właściwej stronie
- Jeżeli  $\xi_i > 1$ , punkt po niewłaściwej stronie hiperpłaszczyzny i wystąpi błąd klasyfikacji
- Modyfikacja wymagań dla wektorów nośnych

$$b_{i1} \quad \mathbf{w} \cdot \mathbf{x} + \mathbf{b} = 1 - \xi$$

$$b_{i2} \quad \mathbf{w} \cdot \mathbf{x} + \mathbf{b} = -1 + \xi$$

# Linear SVM – niepełna separowalność



- For all ☺

$$w_1 * x_1 + w_2 * x_2 \geq b + 1$$

- For ☹

$$w_1 * x_1 + w_2 * x_2 \geq b + 1 - \xi$$

...for some positive  $\xi$

- Task: Maximize the margin *and* minimise training errors

$$\frac{\|\mathbf{w}\|^2}{2} \xrightarrow{\text{BECOMES}} \frac{\|\mathbf{w}\|^2}{2} + C\left(\sum_i \xi_i\right)$$

minimize

minimize

# SVM z dodatkowymi zmiennymi

- Jak przededefiniować sformułowanie? Z dodatkowymi zmiennymi dopełniającym oraz kosztem błędu na danych uczących

- Minimalizuj wyrażenie: 
$$L(w) = \frac{\|\vec{w}\|^2}{2} + C \left( \sum_{i=1}^N \xi_i^k \right)$$

- z ograniczeniami:

- $$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 - \xi_i \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 + \xi_i \end{cases}$$

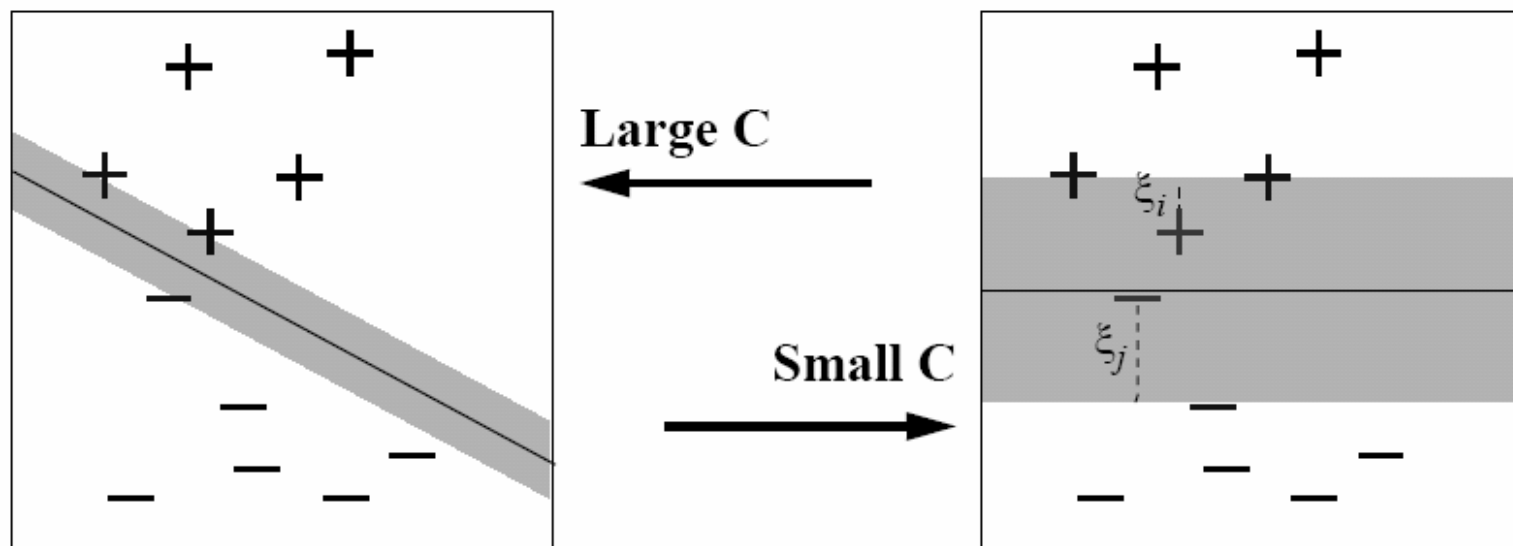
- Drugi czynnik odpowiada za ew. błędy testowania (górne oszacowanie tych błędów)
- Parametr C ocena straty związanej z każdym błędnie klasyfikowanym punktem dla które  $\xi > 0$
- Przetarg „szeroki margines” to dużo błędów i odwrotnie

# Controlling Soft-Margin Separation

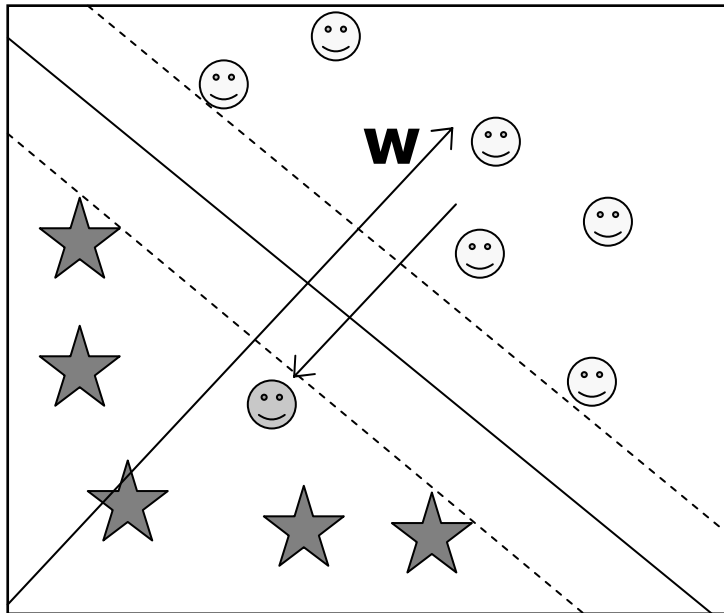
$$\text{Soft Margin: minimize } P(\vec{w}, b, \vec{\xi}) = \frac{1}{2} \vec{w} \cdot \vec{w} + C \sum_{i=1}^n \xi_i$$

$$\text{s. t. } y_i[\vec{w} \cdot \vec{x}_i + b] \geq 1 - \xi_i \text{ and } \xi_i \geq 0$$

- $\sum \xi_i$  is an upper bound on the number of training errors.
- $C$  is a parameter that controls trade-off between margin and error.



# Rozwiązanie problemu



- Ponownie dojdziemy do dualnego problemu:

$$\text{Max: } W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \bullet \mathbf{x}_j)$$

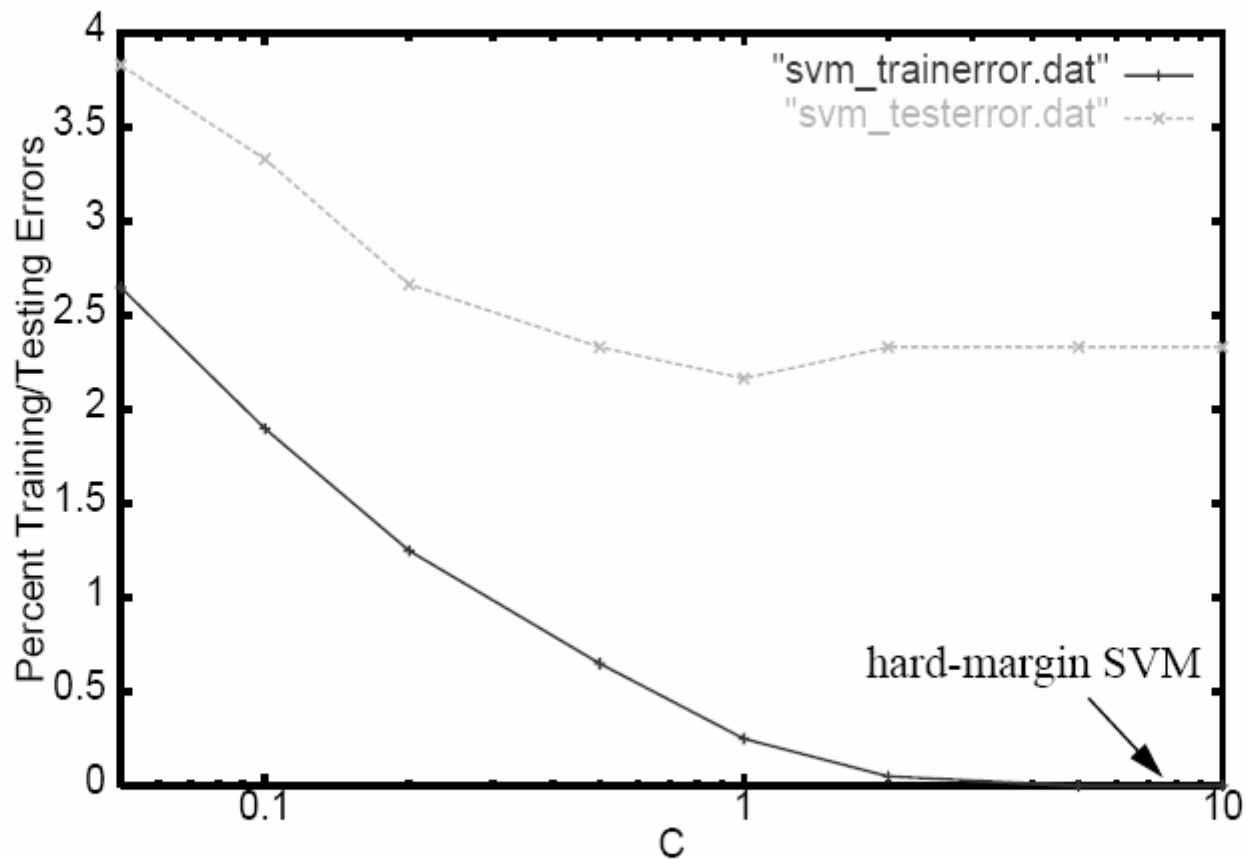
ogranicz:

- (1)  $0 \leq \alpha_i \leq C, \quad \forall i$
- (2)  $\sum_{i=1}^m \alpha_i y_i = 0$

Programowanie kwadratowe (QP) :  
 trudności rozwiązania →  
 przeformułuj problem

Globalne max  $\alpha_i$  może być  
 odnaleziono

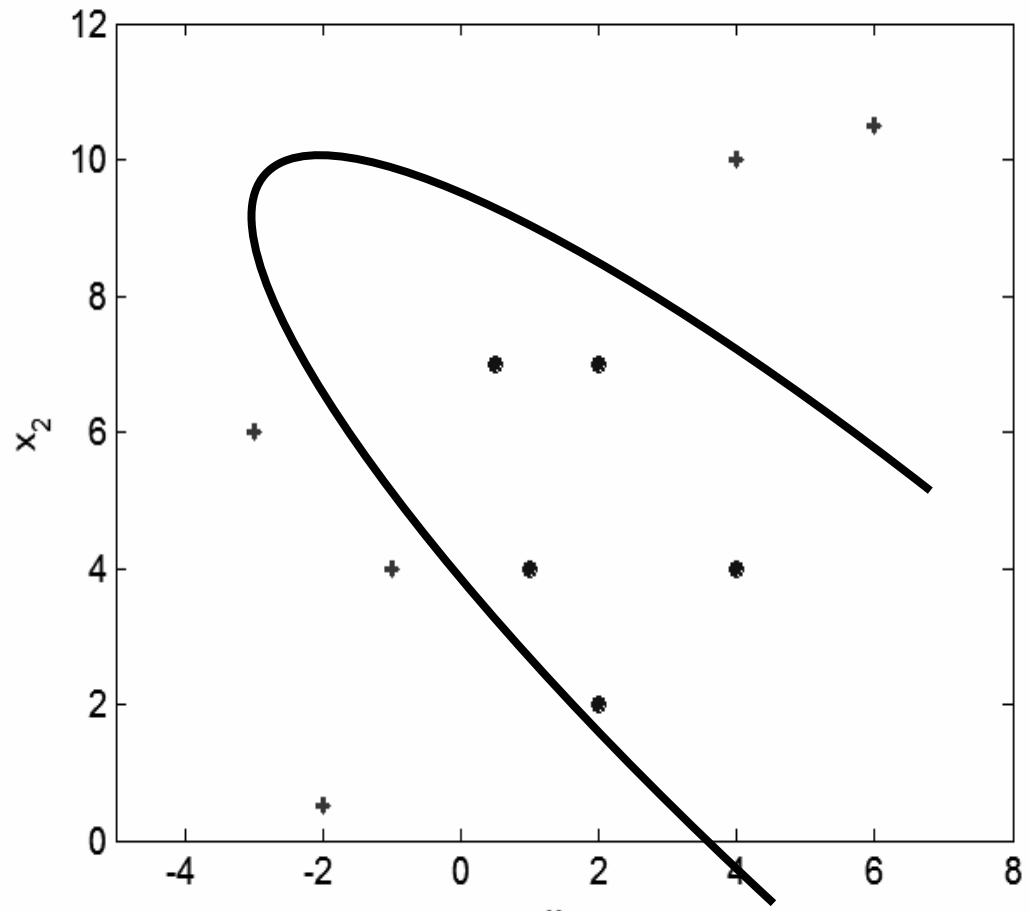
## Example Reuters “acq”: Varying C



**Observation:** Typically no local optima, but not necessarily...

# Nonlinear Support Vector Machines

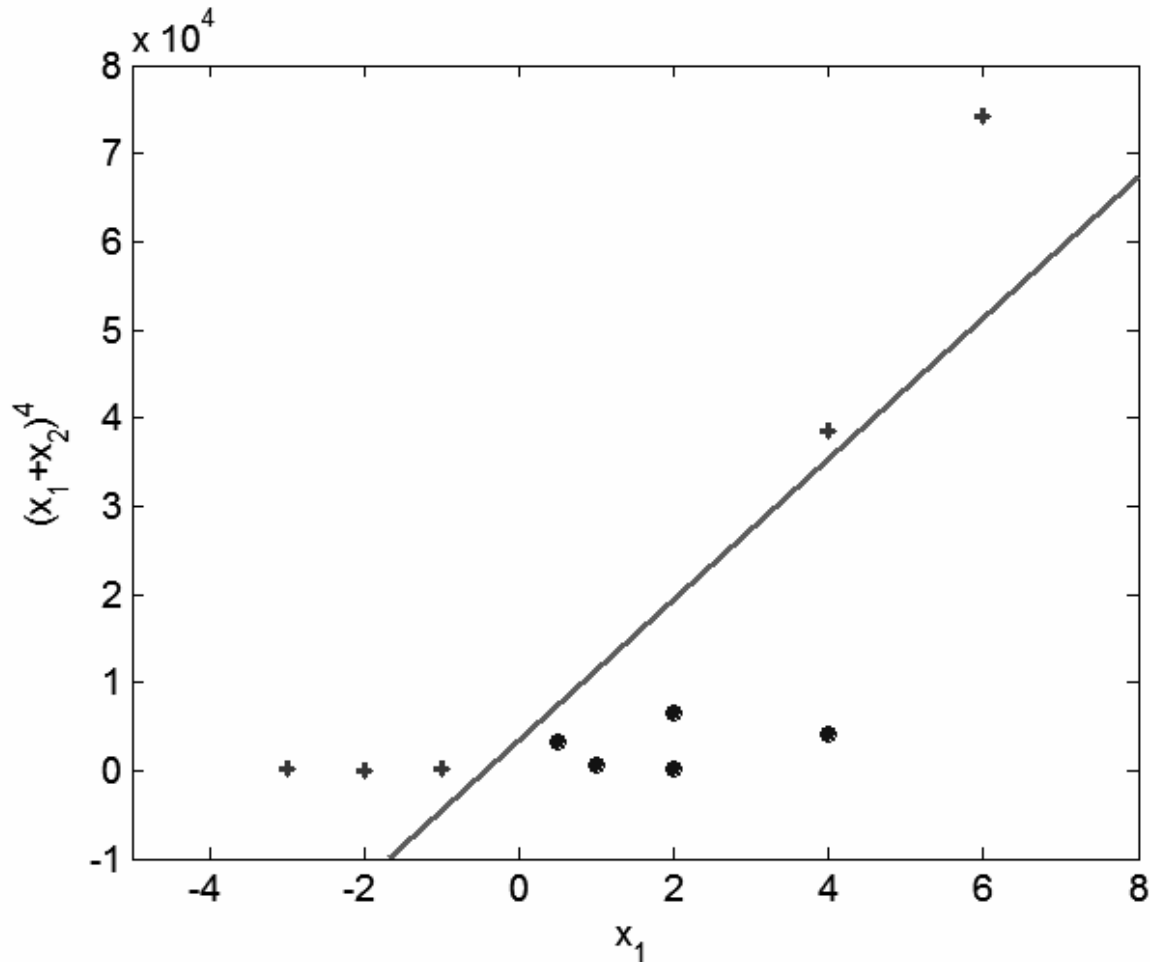
- Co zrobić gdy próby uczące powinny być nieliniowo separowalne?





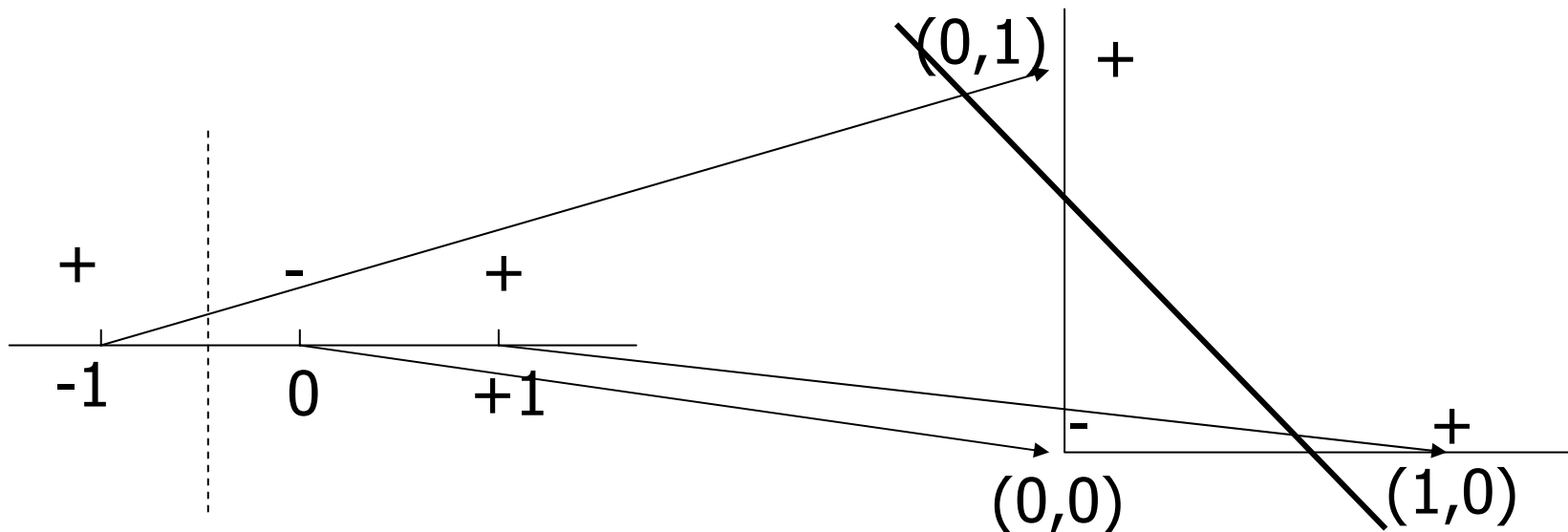
# Nonlinear Support Vector Machines

- Transformacja do wysoce wielowymiarowej przestrzeni

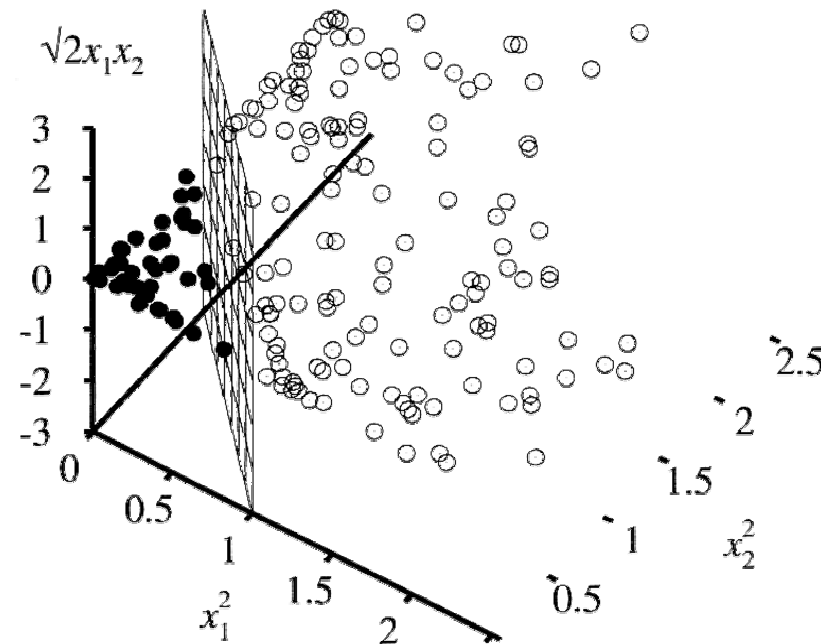
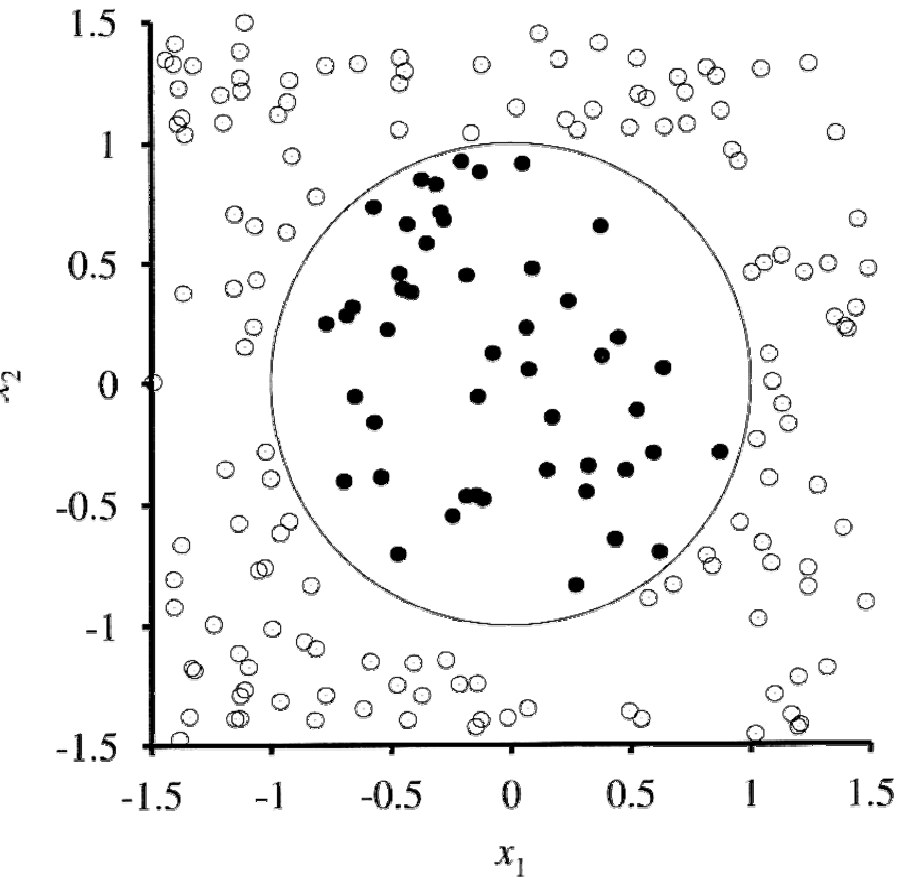


# SVM – Transformacje

- Przykład transformacji  $1D \rightarrow 2D$
- Projekcja danych oryginalnych  $x \in R^p$  w nową  $m > p$  wielowymiarową przestrzeń, w której z dużym prawdopodobieństwem będą separowalne liniowo (Twierdzenia matem. np. Covera)
- Przykład przekształcenia wielomianowe wyższego stopnia gdzie do zmiennych  $x$  dołącza się ich  $p$ -te potęgi oraz iloczyny mieszane.



# Przykład transformacji wielomianowej



# Przykład transformacji wielomianowej

- Oryginalna funkcja celu

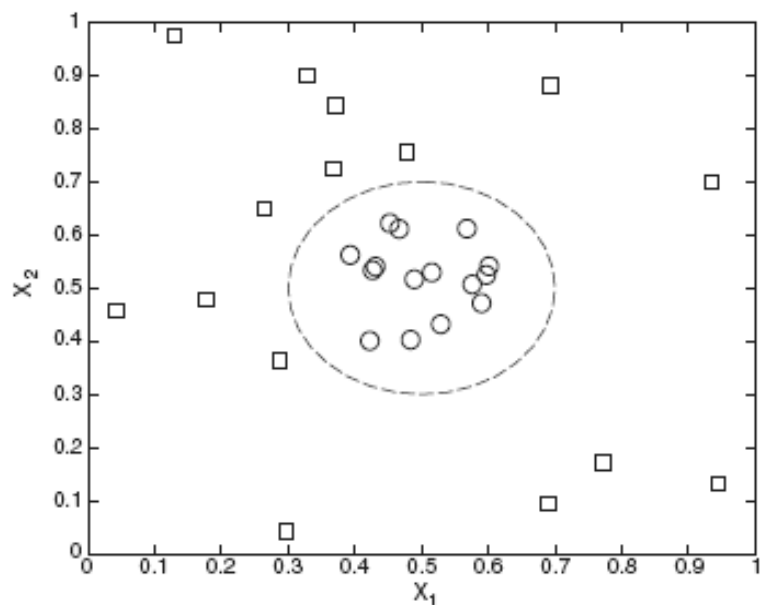
$$y(x_1, x_2) = \begin{cases} 1 & \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} > 0.2 \\ -1 & \text{otherwise} \end{cases}$$

- Transformacja  $\Phi : (x_1, x_2) \rightarrow (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1)$
- Poszukujemy parametrów

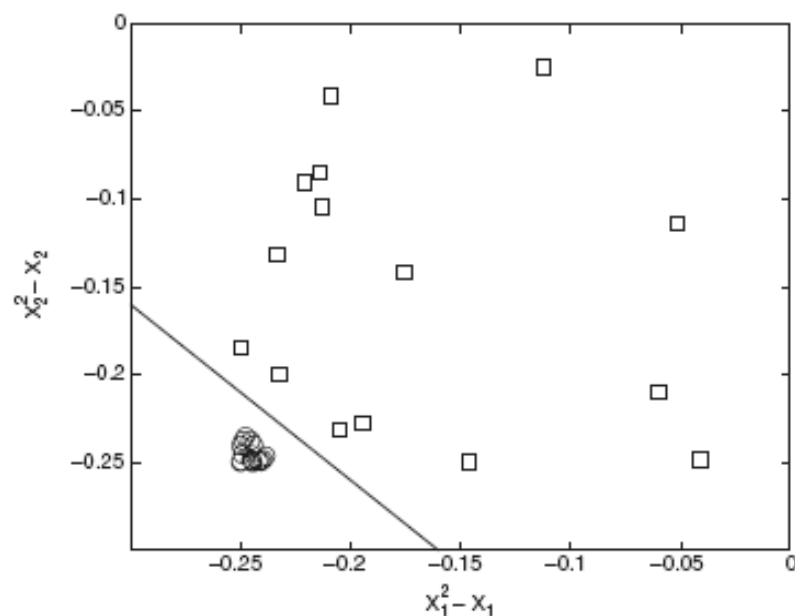
$$w_4 x_1^2 + w_3 x_2^2 + w_2 \sqrt{2}x_1 + w_1 \sqrt{2}x_2 + w_0 = 0$$

- Rozwiązanie

$$x_1^2 - x_1 + x_2^2 - x_2 = -0.46$$



(a) Decision boundary in the original two-dimensional space.

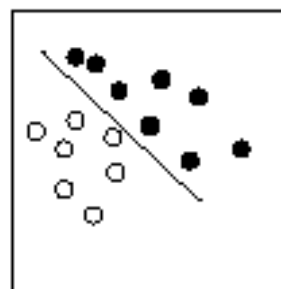


(b) Decision boundary in the transformed space.

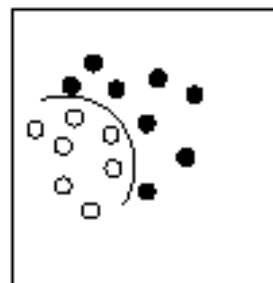
**Figure 5.28.** Classifying data with a nonlinear decision boundary.

# Kilka definicji

- Przykłady danych do klasyfikacji

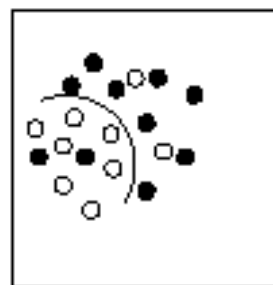
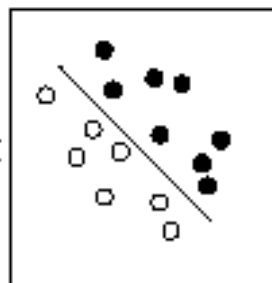


separable  
linear



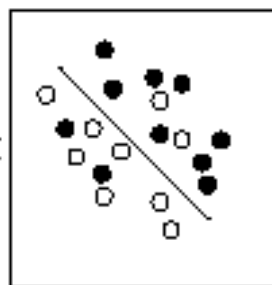
separable  
nonlinear

$\Phi$   
nonlinear  
map



nonseparable  
nonlinear

$\Phi$   
nonlinear  
map



# Model nieliniowy SVM

- Funkcja decyzyjna po przekształceniu  $g(\mathbf{x}) = \mathbf{w} \cdot \Phi(\mathbf{x}) + b$
- Zasady klasyfikacji
$$\begin{array}{ll} +1 & g(\mathbf{x}) > 0 \\ -1 & g(\mathbf{x}) < 0 \end{array}$$
- Sformułowanie problemu nieliniowego SVM

$$\min_{\mathbf{w}} = \frac{\|\mathbf{w}\|^2}{2} \quad y_i(\mathbf{w} \cdot \Phi(x_i) + b) \geq 1 \quad i = 1, 2, \dots, N$$

- Podobnie jak poprzednio optymalizujemy funkcje z mnożnikami Lagrange'a

$$\sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j))$$

- Funkcja klasyfikująca

$$f(x) = \text{sign}\left(\sum_{i=1}^N \alpha_i y_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}) + b\right)$$

# Curse of dimensionality and ...

- Oblicz  $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$
- **Problem:** Trudne obliczeniowo do wykonania!
- Wiele parametrów do oszacowania - wielomian stopnia  $p$  dla  $N$  atrybutów w oryginalnej przestrzeni prowadzi do  $O(N^P)$  atrybutów w nowej rozszerzonej  $F$  feature space
- Skorzystaj z dot product (iloczynu skalarnego) na wektorach wejściowych jako miary podobieństwa wektorów
- Iloczyn  $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$  może być odniesiony do podobieństwa wektorów  $\mathbf{x}_i \cdot \mathbf{x}_j$  w transformowanej rozszerzonej przestrzeni
- Idea kerneli (funkcji jądrowych)
  - Proste funkcje  $K$  dwóch argumentów wektorowych pozwalają obliczyć wartość iloczynu skalarnego w rozszerzonej przestrzeni



# Co to są funkcje jądrowe (Kernel function)

- Wywodzą się z badań liniowych przestrzeni wektorowych, przestrzeni Hilberta, Banacha
- Intuicyjnie są to stosunkowo proste symetryczne  $K(\mathbf{x}_i, \mathbf{x}_j)$  zależne od odległości między  $\mathbf{x}_i$  i  $\mathbf{x}_j$  które spełniają pewne wymagania matematyczne.

$$K(u) \geq 0, \int K(u) du = 1, \sigma_K^2 = \int u K(u) du > 0$$

Niech  $X$  oznacza niepusty zbiór wektorów danych.

- Funkcję  $\psi : X \times X \mapsto R$  nazywamy dodatnio określonym kernelem (p.d. Mercer kernel), wtedy i tylko wtedy gdy

$$\sum_{i=1}^n \sum_{k=1}^n c_j c_k \psi(\mathbf{x}_j, \mathbf{x}_k) \geq 0$$

dla wszystkich  $n \in N$ ,  $\mathbf{x}_1, \dots, \mathbf{x}_n \subseteq X$ , oraz  $c_1, \dots, c_n \subseteq R$ .

# Wniosek z twierdzenia Mercera

• Twierdzenie. Niech  $K(x, y)$  oznacza funkcję symetryczną dwóch wektorów będącą kernelem, taką że  $\forall x, y \in X, X \subseteq R$ . Wtedy możemy określić przekształcenie  $\phi : X \mapsto \mathcal{F}$ , takie że

$$K(x, y) = \phi(x) \cdot \phi(y).$$

Przestrzeń  $\mathcal{F}$ , do której następuje mapowanie, jest nazywana przestrzenią zmiennych przekształconych (Feature space).

- Własność podstawą tzw. triku kernelowego (*kernel trick*) :
- „... map the data into some other scalar product space (feature space)  $F$  by means of a nonlinear mapping like the above, and perform the linear algorithm (like decision boundary for 2 classes) in the feature space  $F$ . In many cases the mapping  $\Phi$  cannot be explicitly computed, due to the high-dimensionality of  $F$ . But this is not an obstacle, when the decision function requires the evaluation of scalar products  $\Phi(x) \cdot \Phi(y)$ , and not the pattern  $\Phi(x)$  in explicit form.”[Camastra]
- „every dot product is replaced by a non-linear kernel function. „

# Typowo stosowane jądra

## Dopuszczalne typy jąder związane z SVM

Normalne (Gaussowskie)	$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left\{-\frac{(\mathbf{x}_i - \mathbf{x}_j)^2}{2\sigma^2}\right\}$
Wielomianowe	$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + d)^p$
sigmoidalne	$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\kappa \mathbf{x}_i \cdot \mathbf{x}_j - \delta)$

Konstruujemy wektory zmiennych rozszerzonych za pomocą przekształcenia wielomianowego stopnia drugiego ( $p=2$ ):

$$\Phi(\mathbf{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

$$\Phi(\mathbf{y}) = (1, \sqrt{2}y_1, \sqrt{2}y_2, y_1^2, y_2^2, \sqrt{2}y_1y_2).$$

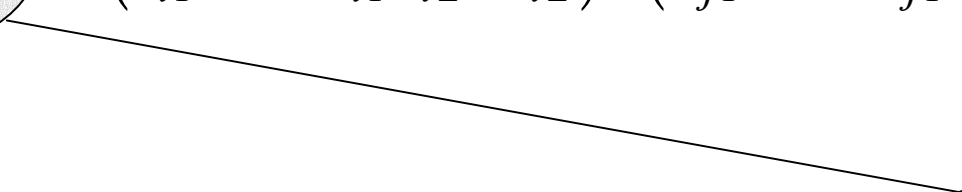
Wtedy okazuje się że iloczyn skalarny w przestrzeni zmiennych przekształconych można wyrazić jako funkcję iloczynu skalarnego zmiennych obserwowanych w  $R^d$ :

$$(\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})) = (1 + \mathbf{x} \cdot \mathbf{y})^2.$$

# SVM: the kernel trick

Przykład prostego przekształcenia wielomianowego

The kernel trick:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^2 = (x_{i1}^2, \sqrt{2}x_{i1}x_{i2}, x_{i2}^2) \cdot (x_{j1}^2, \sqrt{2}x_{j1}x_{j2}, x_{j2}^2) \\ = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$
A line with an arrow points from the circled term  $(\mathbf{x}_i \cdot \mathbf{x}_j)^2$  in the equation above to the circled term  $K(\mathbf{x}_i \cdot \mathbf{x}_j)$  in the equation below.

Original optimization function:

$$\sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i \cdot \mathbf{x}_j)$$

Nie musimy znać funkcji  $\Phi$ , wystarczy znać jądro (kernel)  
i można pracować w nowej przestrzeni

# Funkcja decyzyjna

---

- Wykorzystanie funkcji jądrowych

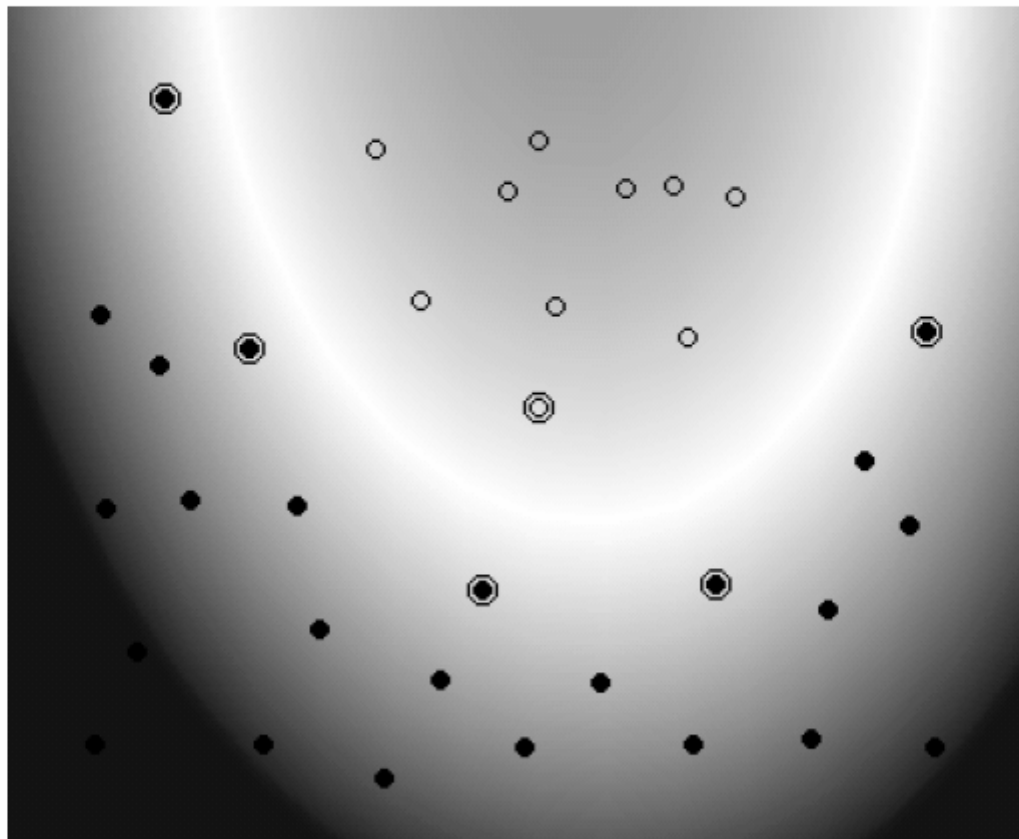
$$f(\mathbf{x}) = \begin{matrix} \text{sign}\left(\sum_{i=1}^N \alpha_i y_i \Phi(\mathbf{x}_i) \Phi(\mathbf{x}) + b\right) \\ \text{sign}\left(\sum_{i=1}^N \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right) \end{matrix}$$

- Model klasyfikacji binarnej rozszerza się na zagadnienie wieloklasowe  $K > 2$ 
  - Specyficzne konstrukcje złożone:
    - one-versus-all
    - Pairwise classification (Hastie,...)

## Example: SVM with Polynomial of Degree 2

Kernel:  $K(\vec{x}_i, \vec{x}_j) = [\vec{x}_i \cdot \vec{x}_j + 1]^2$

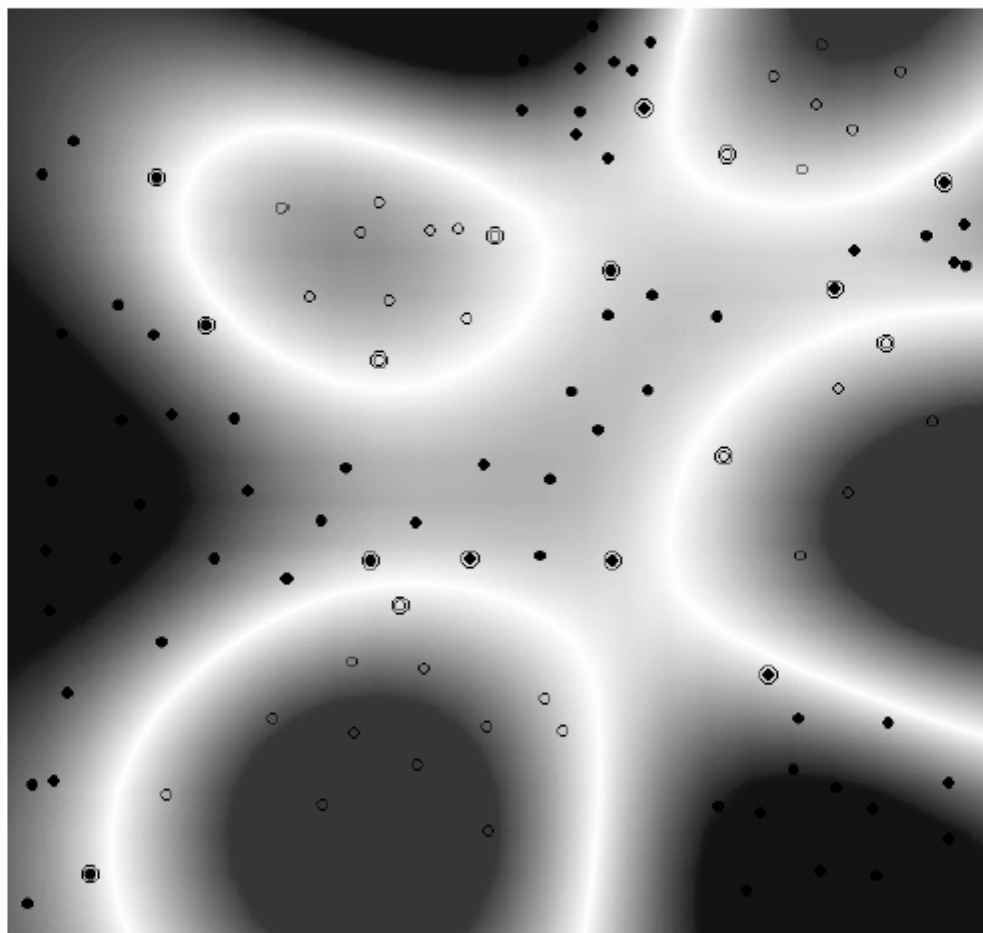
plot by Bell SVM applet



# Example: SVM with RBF-Kernel

Kernel:  $K(\vec{x}_i, \vec{x}_j) = \exp(-|\vec{x}_i - \vec{x}_j|^2 / \sigma^2)$

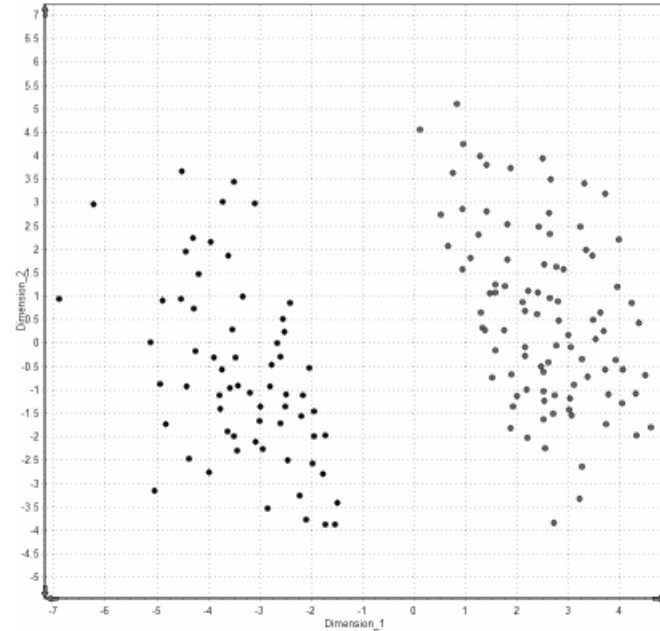
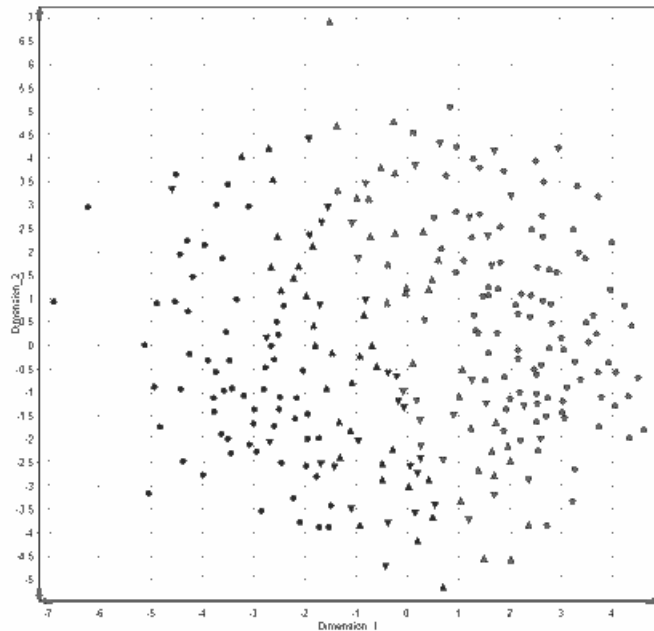
plot by Bell SVM applet



# Example 2: Cleveland heart data

Left: 2D MDS features, linear SVM,  $C=1$ , acc. 81.9%

Right: support vectors removed, margin is clear.



Gaussian kernel,  $C=10000$ , 10xCV, 100% train,  $79.3 \pm 7.8$  test

Gaussian kernel,  $C=1$ , 10xCV, 93.8% train,  $82.6 \pm 8.0$  test

Auto  $C=32$  and Gaussian dispersion 0.004: about  $84.4 \pm 5.1$  on test



# Przykładowe zastosowania

---

Można się zapoznać z listą:

<http://www.clopinet.com/isabelle/Projects/SVM/applist.html>

A few interesting applications, with highly competitive results:

- On-line Handwriting Recognition, zip codes
- 3D object recognition
- Stock forecasting
- Intrusion Detection Systems (IDSs)
- Image classification
- Detecting Steganography in digital images
- Medical applications: diagnostics, survival rates ...
- Technical: Combustion Engine Knock Detection
- Elementary Particle Identification in High Energy Physics
- Bioinformatics: protein properties, genomics, microarrays
- Information retrieval, text categorization

# Trochę historii

---

- Wczesne lata sześćdziesiąte – została opracowana metoda “support vectors” w celu konstruowania hiperpłaszczyzn do rozpoznawania obrazu ( Vapnik i Lerner 1963, Vapnik i Czervonenkis 1964) – liniowa SVM.
- Początek lat 1990-siātych: uogólnienie metody pozwalające na konstruowanie nieliniowych funkcji separujących (Boser 1992, Cortes i Vapnik 1995).
- 1995: dalsze rozszerzenie pozwalające otrzymać estymację funkcji ciągłej na wyjściu – regresja (Vapnik 1995).

# Kilka zagadn. efektywnego stosowania SVM

---

- Normalizuj sygnały wejściowe
- Dobrze wybierz wartość  $C$
- Wybór funkcji jądrowej
- Uogólnienia dla problemów wieloklasowych
- ... co jeszcze?
- Na ile są skuteczne SVM w analizie danych niezrównoważonych

# Parę uwag podsumowujących

---

- Dane odwzorowane (przy pomocy funkcji jądrowych) w nową przestrzeń cech – silna przewaga nad innymi metodami
- W nowej przestrzeni dane powinny być liniowo separowalne
- W porównaniu do innych podejść wielowymiarowość przekształcenia jest „rozwiązana” przez trick kernelowy
- Pośrednio ogranicza się niebezpieczeństwo przeuczenia
- Teoretycznie poszukują minimum globalnego a nie lokalnego (jak podejścia heurystyczne – MLP)
- Ograniczenia
  - Dobór parametrów
  - Skrajne podejście „black box”

# Mocne strony SVM

---

Stopień skomplikowania/pojemność jest niezależna od liczby wymiarów.

Bardzo dobra podbudowa statystyczno-teoretyczna

Znajdowanie minimum globalnego. Minimalizujemy funkcję kwadratową co gwarantuje zawsze znalezienie minimum.

Algorytm jest wydajny i SVM generuje prawie optymalny klasyfikator. Nie jest też czuły na przetrenowanie.

Dobre uogólnianie dzięki wielowymiarowej “feature space”.

**Najważniejsze: poprzez użycie odpowiedniej funkcji jądra SVM bardzo duża skuteczność w praktyce**

# Słabe strony SVM

---

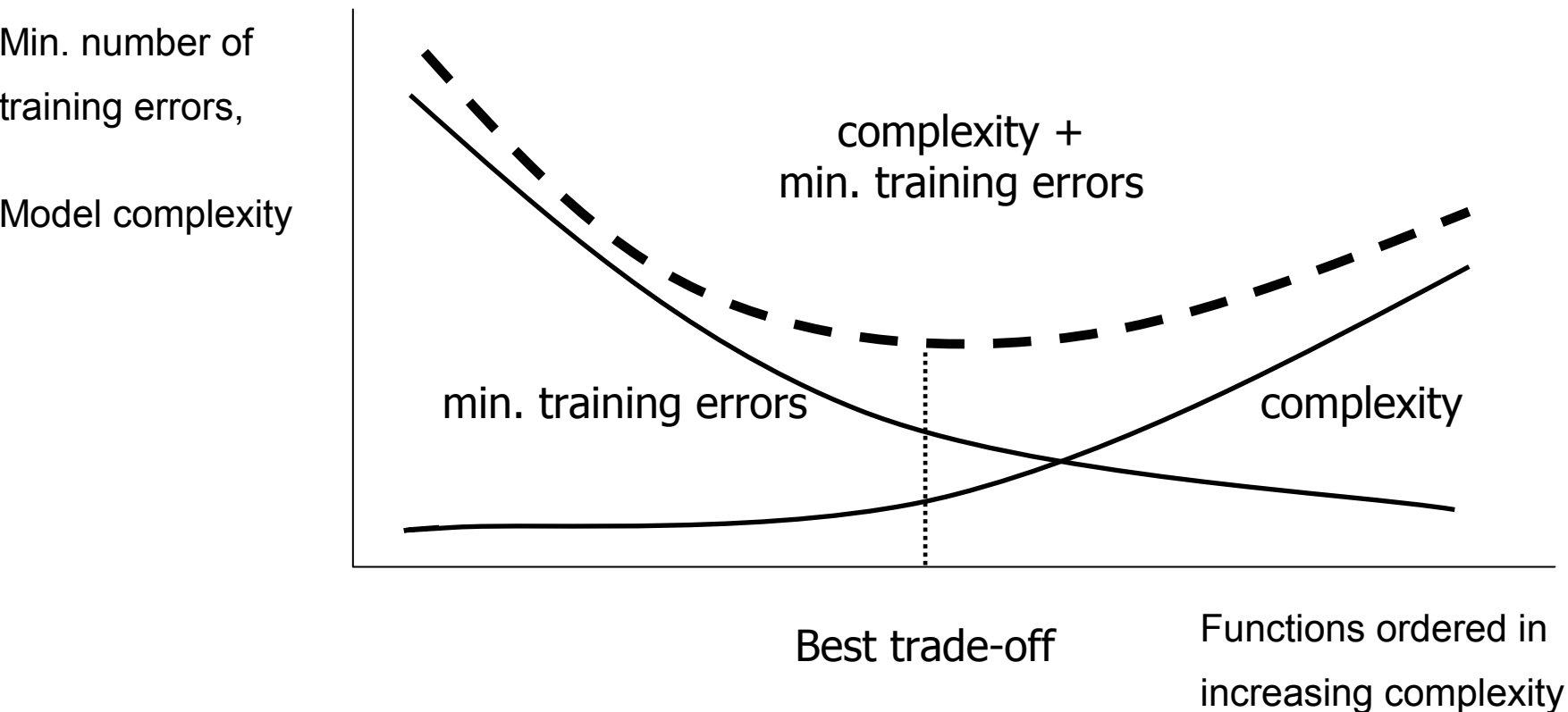
Powolny trening – minimalizacja funkcji, szczególnie dokuczliwy przy dużej ilości danych użytych do treningu.

Rozwiązania też są skomplikowane (normalnie >60% wektorów użytych do nauki staje się wektorami wspierającymi), szczególnie dla dużych ilości danych.

*Przykład (Haykin): poprawa o 1.5% ponad wynik osiągnięty przez MLP. Ale MLP używał 2 ukrytych węzłów, a SVM 285 wektorów.*

Trudno dodać własną wiedzę (prior knowledge)

# Przetarg między złożonością modelu a minimalizacją błędów



# Odnośniki literaturowe

---

- T.Hastie, R.Tibshirani, J.Friedman: The Elements of Statistical Learning. Springer → poszukaj wersji elektronicznej pdf
- J.Koronacki, J.Ćwik: Statystyczne systemy uczące się (rozdz. 6)
- M.Krzyśko, W.Wołyński, T.Górecki, M.Skorzybut: Systemy uczące się.
- S.Osowski: Sieci neuronowe w przetwarzaniu informacji.



# Inne materiały - internet

---

- A.Bartkowiak: Wykłady nt. Sieci Neuronowych: w11 Kernele, siecie SVM i sieci GDA.
  - <http://www.ii.uni.wroc.pl/~aba/>
- W.Duch: wykłady nt. Computational Intelligence
  - [http://www.fizyka.umk.pl/~duch/Wyklady/NN\\_plan.html](http://www.fizyka.umk.pl/~duch/Wyklady/NN_plan.html)
- Angielska wersja Wikipedii
- Thorsten Joachims: Support Vector and Kernel Methods - SIGIR 2003 Tutorial

# SVM Related Links

---

- SVM Website
  - <http://www.kernel-machines.org/>
- Representative implementations
  - **LIBSVM**: an efficient implementation of SVM, multi-class classifications, nu-SVM, one-class SVM, including also various interfaces with java, python, etc.
  - SVM-light: simpler but performance is not better than LIBSVM, support only binary classification and only C language
  - SVM-torch: another recent implementation also written in C.

# Inne odnośniki do literatury anglojęzycznej

---

- “Statistical Learning Theory” by Vapnik: extremely hard to understand, containing many errors too.
- C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Knowledge Discovery and Data Mining*, 2(2), 1998.
  - Better than the Vapnik’s book, but still written too hard for introduction, and the examples are so not-intuitive
- The book “An Introduction to Support Vector Machines” by N. Cristianini and J. Shawe-Taylor
  - Also written hard for introduction, but the explanation about the mercer’s theorem is better than above literatures
- The neural network book by Haykins
  - Contains one nice chapter of SVM introduction